



INTEL
OpenSource
TECHNOLOGY CENTER

IPv6 for Developers used to IPv4

Thiago Macieira

Embedded Linux Conference / OpenIoT Summit – April, 2016

Timeline

1993

IPng formed

1995

First IPv6 RFC



1999/2000

Thiago begins contributing to
OSS (an IPv6-capable browser)

?

IANA IPv4 exhaustion

Timeline

1993

IPng formed

1995

First IPv6 RFC



1999/2000

Thiago begins contributing to
OSS (an IPv6-capable browser)

2011-01-31

IANA IPv4 exhaustion



Overview of IPv6

API

Things you can do with IPv6

Wait, what happened to IPv5?

Our last, best hope for addressing

Comparison to IPv4

	IPv4	IPv6
Address size	32 bits	128 bits
Multicasting	Optional	Mandatory
Minimum MTU	68 octets	1280 octets
Maximum packet size	65,535 octets	4,294,967,295 octets
Fragmentation	By routers	At origin
Privacy extensions	No	Yes
LL address resolution protocol	ARP	IPv6 (ICMPv6)

Some things invented for IPv6 made it to IPv4

- **Link-local addresses**

169.254.x.y/16

- **Anycasting**

192.88.99.1

- **IPsec**

An IPv6 address

- **IPv4:**

198.51.100.1

- **IPv6:**

2001:0DB8:AC10:FE01:0000:0000:0000:0000

2001:db8:ac10:fe01::

9R}vS@^\$5zhx`UX_mr)^
(RFC 1924 compact form)

Localhost and anyhost

- **IPv4:**

0.0.0.0/8
127.0.0.1/8

- **IPv6:**

0000:0000:0000:0000:0000:0000:0000:0000/128 → **::/128**

0000:0000:0000:0000:0000:0000:0000:0001/128 → **::1/128**

Stateless address auto-configuration (SLAAC)

- **Enables hosts to communicate without DHCP servers**
- **If a router is present:**
 - Can configure global addresses statelessly
 - Can query DHCPv6 server for extra information (DNS servers, NTP servers, etc.) or more IPv6 addresses

SLAAC Overview

MAC address: 00:01:5E:7C:49:F8



Modified EUI-64: 00:01:5E:FF:FE:7C:49:F8



IPv6 interface identifier: ::200:5eff:fe7c:49f8



Link-local prefix: fe80::/64



fe80::200:5eff:fe7c:49f8

Obtained from router: 2001:db8:ac10:fe01::/64



2001:db8:ac10:fe01:200:5eff:fe7c:49f8

What about my privacy?

- **Temporary addresses**

- RFC 4941
- Randomly generated
- Rotated after a time
- On Linux, set value 2 in `sysctl net.ipv6.conf.ifname.use_tempaddr`
- Also supported by NetworkManager

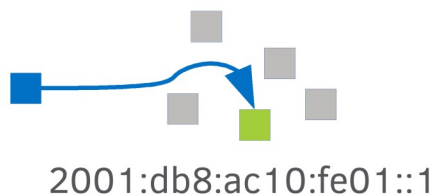
- **Stable but opaque addresses**

- RFC 7217
- Suggestion: Result of a pseudorandom function (e.g., SHA-1)
- Supported in the Linux kernel since 4.1 (`net.ipv6.conf.ifname.stable_secret`)
- Supported by `dhcpcd` 6.4 & NM 1.2

The “casts”

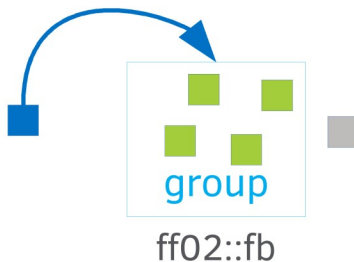
- **Unicast**

- One to one



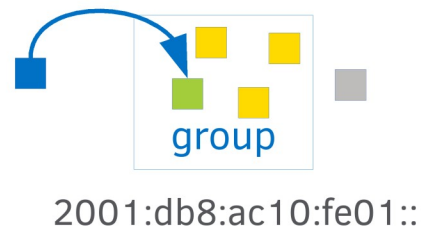
- **Multicast**

- One to many
(all in a group)



- **Anycast**

- One to any
(one of a group)





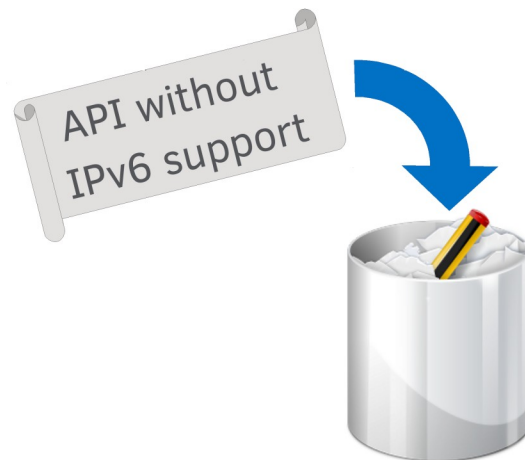
Overview of IPv6

API

Things you can do with IPv6

What you need to know

- **Don't assume anything about the address!**
- **Use high-level API that supports IPv6**
 - libcurl, libsoup, Qt; Python libraries, etc.
- **Use the IPv6 API, always**
 - IPv6 sockets can talk to IPv4



For example, URLs

- URL is

scheme://host:port/path/?query#fragment

- Construct the URL for

```
scheme  http
host    2001:db8:ac10:fe01:200:5eff:fe7c:49f8
port    80
path    /
```

http://[2001:db8:ac10:fe01:200:5eff:fe7c:49f8]:80/

A socket address (TCP, UDP, SCTP, etc.)

sockaddr_in

```
struct sockaddr_in
{
    sa_family_t sin_family;
    in_port_t sin_port; /* Port number. */
    struct in_addr sin_addr; /* Internet address. */
};
```

sockaddr_in6

```
struct sockaddr_in6
{
    sa_family_t sin6_family;
    in_port_t sin6_port; /* Transport layer port # */
    uint32_t sin6_flowinfo; /* IPv6 flow information */
    struct in6_addr sin6_addr; /* IPv6 address */
    uint32_t sin6_scope_id; /* IPv6 scope-id */
};
```

sockaddr_storage

- Big enough for all your addresses

Resolving an address: getaddrinfo()

```
#include <arpa/inet.h>      /* for inet_ntop/inet_pton */
#include <netdb.h>           /* getaddrinfo */
#include <stdio.h>
#include <string.h>

static int use_addrinfo(const struct addrinfo *ai);
int main(int argc, char **argv)
{
    struct addrinfo hints, *result, *p;
    memset(&hints, 0, sizeof(hints));
    hints.ai_family = AF_UNSPEC;      /* ask for any address family */
    hints.ai_flags = AI_ADDRCONFIG;  /* only return IPv6 if the host has IPv6 */
    hints.ai_flags |= AI_CANONNAME;  /* request the host's canonical name */
    // hints.ai_flags |= AI_PASSIVE; /* return address suitable for bind() */
    hints.ai_socktype = SOCK_STREAM; /* ask for TCP sockets */

    int ret = getaddrinfo(argv[1], "http", &hints, &result);
    if (ret) {
        fprintf(stderr, "Failed to resolve %s: %s\n", argv[1], gai_strerror(ret));
    } else {
        for (p = result; p; p = p->ai_next) {
            int fd = use_addrinfo(p);
            if (fd != -1)
                break;
        }
        freeaddrinfo(result);
    }
    return ret;
}
```

Reversing the resolution: getnameinfo()

```
static int use_addrinfo(const struct addrinfo *ai)
{
    char buf[NI_MAXHOST];
    getnameinfo(ai->ai_addr, ai->ai_addrlen,
                buf, sizeof(buf),
                NULL, 0,          // no port number
                NI_NUMERICHOST);

    printf("%s: %s %s\n",
          ai->ai_family == AF_INET6 ? "IPv6" : "IPv4",
          buf,
          ai->ai_canonname ? ai->ai_canonname : "");

    return -1;
}
```

```
$ ./a.out www.kame.net
IPv4: 203.178.141.194 orange.kame.net
IPv6: 2001:200:dff:fff1:216:3eff:feb1:44d7
$ ./a.out chat.freenode.net
IPv4: 174.143.119.91 chat.freenode.net
IPv4: 193.219.128.49
IPv4: 91.217.189.42
IPv4: 192.186.157.43
IPv4: 195.154.200.232
IPv4: 185.30.166.38
IPv4: 82.96.64.4
IPv4: 193.10.255.100
IPv4: 185.30.166.37
IPv4: 83.170.73.249
IPv4: 94.125.182.252
IPv4: 130.239.18.119
IPv4: 84.240.3.129
IPv4: 164.132.77.237
IPv4: 162.213.39.42
IPv6: 2001:778:627f::1:0:49
IPv6: 2001:948:7:7::140
IPv6: 2001:6b0:e:2a18:5054:ff:fe01:8119
IPv6: 2a02:2498:1:3a3:6ef0:49ff:fe44:bc07
IPv6: 2a00:1a28:1100:11::42
IPv6: 2a01:270:0:666f::1

othermachine$ ./a.out www.kame.net
IPv6: 2001:200:dff:fff1:216:3eff:feb1:44d7 orange.kame.net
IPv4: 203.178.141.194
```

Connecting to the host

```
#include <sys/socket.h>
#include <unistd.h>

static int use_addrinfo(const struct addrinfo *ai)
{
    int fd = socket(ai->ai_family, ai->ai_socktype, ai->ai_protocol);
    if (fd == -1)
        return -1;
    if (connect(fd, ai->ai_addr, ai->ai_addrlen) < 0) {
        close(fd);
        return -1;
    }
    //return fd;

    static const char msg[] = "GET / HTTP/1.0\r\n\r\n";
    char buf[256];
    ssize_t n;
    write(fd, msg, strlen(msg));
    while ((n = read(fd, buf, sizeof(buf))) > 0)
        fwrite(buf, n, 1, stdout);
    close(fd);
    return 0;
}
```

Servers: IPv6 \supset IPv4

- Listen on dual-stack

- IPv4 clients can connect just fine
::ffff:192.51.100.1
- Default on Linux
- Can be changed

```
#include <sys/socket.h>
#include <unistd.h>

static int use_addrinfo(const struct addrinfo *ai)
{
    int fd = socket(ai->ai_family, ai->ai_socktype, ai->ai_protocol);
    if (fd == -1)
        return fd;

    if (ai->ai_family == AF_INET6) {
        /* Make sure we're getting dual-stack */
        int on = 1;
        setsockopt(fd, SOL_IPV6, IPV6_V6ONLY, &on, sizeof(on));
    }

    if (bind(fd, ai->ai_addr, ai->ai_addrlen) < 0 ||
        listen(fd, 256) < 0) {
        close(fd);
        return -1;
    }
    return fd;
}
```



Overview of IPv6

API

Things you can do with IPv6

Replace all your broadcast with multicast

- Broadcast is “one-to-everyone”
- Can create your group without registering with IANA
- Variable scopes

Scope	Meaning
0	Reserved
1	Node-local or interface-local
2	Link-local
3	Realm-local
4	Admin-local
5	Site-local
8	Organisation-local
e	Global
f	Reserved

More control over packet

- **Advanced Socket API interface (RFC 3542)**

- Ancillary data in `recvmsg` and `sendmsg`
- `IPV6_RECVPKTINFO` (`setsockopt`) / `IPV6_PKTINFO` (control message)

- **Use-case examples:**

What IP address was this UDP datagram addressed to? Was it unicast or was it multicast?

Need to send this UDP datagram on a specific network interface.

6LoWPAN

- **IPv6 over Low-power Wireless Personal Area Network**
 - For IEEE 802.15.4 networks
 - IPv6 API maps 1:1 with radio packets
- **Used by the Thread Group**



Thiago Macieira

thiago.macieira@intel.com

<http://google.com/+ThiagoMacieira>